

TUTORIAL/ASISTENTE PARA EL APRENDIZAJE DE TÉCNICAS BÁSICAS DE DISEÑO DE CIRCUITOS DIGITALES

Nieto Nieto, Luis Miguel¹ y Jareño Oliver, José Simón²

¹ Dpto. de Electrónica. E.U.P. de Linares. Universidad de Jaén

² 2º I.T.T. Esp. Telemática. E.U.P. de Linares, Universidad de Jaén
C/ Alfonso X El Sabio, 28. C.P: 23700 Linares (Jaén)

Tel.: 953-649555. Fax: 953-649508. Email: lmnieto2@ujaen.es

RESUMEN

En el diseño de circuitos digitales de lógica cableada se utilizan ciertos procedimientos muy sistemáticos de obligado conocimiento para el estudiante de estas materias, incluidas, por ejemplo, en asignaturas de primer ciclo de Ingeniería de Telecomunicaciones o de Informática. El programa que se presenta pretende ilustrar y automatizar la aplicación de los mismos, sirviendo como herramienta de cálculo y de ayuda al estudio o en la preparación de ejercicios.

1. INTRODUCCION

El presente trabajo ha estado inicialmente dirigido a alumnos de la asignatura *Electrónica Digital*, integrada en los estudios de Ingeniería Técnica de Telecomunicaciones e impartida en la E.U.P. de Linares (Universidad de Jaén). En dicha asignatura se tratan, entre otros, procedimientos básicos para el diseño de circuitos digitales combinacionales y secuenciales basados en dispositivos estándar SSI y MSI. En particular, los métodos para la simplificación funcional o para el diseño de la lógica de excitación de sistemas secuenciales síncronos son de naturaleza sistemática y han sido integrados en software comercial dedicado a la programación de PLDs, por ejemplo.

Sin embargo, aunque para el alumno es obligado el conocimiento de dichos métodos, el software mencionado no está orientado a la didáctica de los mismos. Igual sucede con simuladores de circuitos como el conocido PSpice, utilizado en las Prácticas de Electrónica Digital impartidas en nuestra Escuela, y que ha supuesto una valiosa ayuda al estudio al

facilitar la *comprobación* funcional de los diseños de una manera rápida y fiable. Por ello, en el ciclo de desarrollo de cada práctica, el simulador ocupó la etapa previa al montaje del prototipo físico, no asistiendo directamente en el planteamiento del diseño preliminar.

Con el propósito de cubrir esta parcela se ha preparado el software presentado, que denominamos *HADD*, consistente en una herramienta de cálculo de utilidad para *plantear* tales diseños, que además trata de *ilustrar* cómo se utilizan las técnicas implicadas.

2. ASPECTOS GENERALES

La actual versión del programa es una aplicación de 32 bits desarrollada con Visual Basic que maneja bases de datos relacionales y consultas SQL. Está destinada a trabajar sobre Windows 95 y un PC basado en Pentium, sin que precise recursos multimedia.

Entre las funciones de cálculo ofrece las siguientes:

- Minimización de una función combinacional de hasta 8 variables, mediante el método tabular de Quine-McCluskey, partiendo de una descripción como suma de minterm [2][3] e indiferencias.
- Minimización conjunta de hasta 6 funciones combinacionales con un máximo de 6 variables, partiendo de una descripción como suma de minterm e indiferencias.
- Implementación de una función combinacional de hasta 6 variables sobre un multiplexor con un máximo de 6 entradas de selección, partiendo de una descripción como suma de minterm e indiferencias.
- Previa codificación de estados, generación de las tablas de excitación para diseño de sistemas secuenciales síncronos basados en biestables (J-K, T o D) o en contadores binarios incrementales con carga en paralelo y capacidad arbitraria.
- Minimización de funciones de excitación generadas para sistemas síncronos, de manera individual o conjunta con las limitaciones arriba indicadas.

También cuenta con ayuda contextual y temas tutoriales, así como ejercicios. Además, para ayudar a la comprensión de los procesos de simplificación, incluye la posibilidad de observar la generación paso a paso de las tablas de búsqueda de implicantes, y observar las adyacencias en mapas de Karnaugh para ilustrar gráficamente el proceso de construcción de adyacencias de ordenes superiores.

3. DESCRIPCIÓN

En la simplificación de funciones lógicas combinacionales se aplica habitualmente el método de los mapas de Karnaugh [1] [2] [3]. Con este método se aprovecha la capacidad que el diseñador disponga para el reconocimiento de patrones gráficos sobre una tabla, cuya especial construcción facilita las agrupaciones de celdas que sean de interés. Su utilización es trivial hasta la dimensión 4 o 5, y es un método rápido de enseñar y aplicar. Sin embargo la dificultad al tratar de reconocer patrones en estos mapas crece con la dimensión de los mismos, volviéndolos prácticamente inaplicables para dimensiones superiores. Además, para

cualquier dimensión, precisa de cierto entrenamiento visual y no es una técnica adecuada para efectuar una simplificación multifuncional conjunta.

La alternativa habitual es el método tabular de Quine-McCluskey [1][2][3], que presenta una sistemática sencilla fácilmente trasportable a un programa de ordenador [1] (en contraposición al método de mapas) y que además permite la simplificación multifuncional, sin limitación en el número de variables o funciones. Sin embargo el método implica repetidas búsquedas exhaustivas y redundantes, convirtiéndolo, en oposición al método de Karnaugh, en un procedimiento tedioso para su aplicación manual.

En el software se han incluido varias herramientas de cálculo que implican como etapa final la simplificación funcional. Además se ha desarrollado como un entorno de ventanas dotado de ayuda contextual, que se ha pretendido sea de manejo intuitivo. Se pretende conseguir que sea suficiente para ayudar al estudiante a asimilar los conceptos tratados, aunque trabaje individualmente, y que además simplifique el trabajo al usuario iniciado.

3.1. Módulo de minimización de funciones combinacionales.

Este módulo se ha diseñado aprovechando las ventajas de ambos métodos de simplificación:

- Implementa, como motor de cálculo, el algoritmo de Quine-McCluskey para minimización individual o conjunta.
- Se sirve de la capacidad informativa de los mapas de Karnaugh para ilustrar gráficamente cómo evoluciona la búsqueda tabular de adyacencias de órdenes crecientes.

Como datos de entrada se deben especificar las funciones combinacionales como suma de productos canónicos, pudiéndose incluir indiferencias. Pueden ser definidas hasta 6 funciones, de las que se realizará la simplificación individual y conjunta para aquellas que sean seleccionadas.

Figura 1: Definición de funciones como suma de minterm más indiferencias

El programa construye una tabla con todas las adyacencias que calcula el método de Quine-McCluskey, los implicantes primos y los esenciales. A la vez genera el mapa de Karnaugh correspondiente a la adyacencia que el usuario seleccione (solo hasta dimensión 5). De este modo el alumno puede aprender simultáneamente ambos métodos, encontrando en el de Quine-McCluskey la *justificación* de las coberturas gráficas observables en los mapas de Karnaugh.

a=0				a=1					
de \ bc	00	01	11	10	de \ bc	00	01	11	10
00	0	1	3	2	00	16	17	19	18
01	4	5	7	6	01	20	21	23	22
11	12	13	15	14	11	28	29	31	30
10	8	9	11	10	10	24	25	27	26

Figura 2: Mapa de Karnaugh de dimensión 5 presentando la adyacencia de orden 3 {0, 2, 8, 10, 16, 18, 24, 26}

3.2. Módulo de generación de funciones de excitación.

Aprovechando el módulo anterior, se incluye una utilidad para generar las funciones de excitación de los elementos de memoria en sistemas secuenciales síncronos, dependiendo de la opción escogida ("J-K", "D", "T" o "Contadores"), realizando además la simplificación de las mismas individual o globalmente, lo que supone una optimización adicional en el diseño que suele soslayarse en cálculos manuales. El usuario tan solo debe codificar los estados, especificar el número de bits de estado y de entrada, decidir el tipo de elemento de memoria a utilizar, elegir entre una minimización individual o conjunta y rellenar la tabla de estado siguiente de acuerdo a esa codificación. El programa se encarga de calcular las tablas de excitación para cada elemento de memoria dependiendo del tipo de biestable escogido, ofreciendo como resultado final las funciones simplificadas. En el caso de la opción "Contador", se generan las funciones que generan los bits de carga del contador binario y la función de control de carga/cuenta (0: carga en paralelo, 1: cuenta incremental) [1].

Los temas de ayuda relacionados hacen hincapié sobre la descomposición de un problema de diseño de sistemas secuenciales síncronos en varios problemas de diseño de sistemas combinatoriales.

3.3. Módulo de implementación sobre multiplexor.

Por último, como alternativa a la implementación como suma de productos de una función lógica combinacional, se integra otra herramienta para realizarla sobre un multiplexor de capacidad arbitraria (hasta 6 entradas de selección de canal).

El usuario especifica la función de la manera habitual, y decide qué variables destinará para selección de canal, quedando las restantes como argumentos de las funciones combinacionales que se aplicarán a los canales de entrada al multiplexor (ver Fig. 3). Sobre estas funciones se realiza también una simplificación automáticamente.

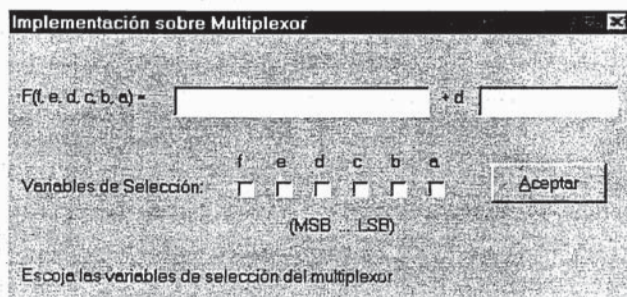


Figura 3: Definición de la función y designación de las variables de selección de canal

4. CONCLUSIONES

Continuando con la línea de mejoras docentes introducidas en la citada asignatura, se presenta un conjunto de herramientas de cálculo que automatizan ciertos procedimientos de diseño de sistemas digitales, y que además incorpora recursos para facilitar la comprensión de tales técnicas. Se ha pretendido ofrecer un entorno de trabajo *amigable*, de uso sencillo dotado de ayuda contextual destinado, principalmente, a estudiantes que se introducen en el diseño digital (v.g. estudiantes de Electrónica Digital de I.T. de Telecomunicaciones), y también dirigido al profesor, dada la ayuda que supone en la preparación de ejercicios.

5. BIBLIOGRAFÍA

- [1] E. Mandado. "Sistemas Electrónicos Digitales". Ed. Marcombo. Agosto, 1990.
- [2] J. F. Wakerly. "Diseño Digital: Principios y Prácticas". Ed. Prentice Hall. 1992.
- [3] F. J. Hill, G. R. Peterson. "Teoría de Conmutación y Diseño Lógico". Ed. Limusa. 1987.
- [4] F. J. Ceballos. "Enciclopedia de Microsoft Visual Basic". Ed. Ra-ma. 1994.